

Guía técnica

Índice:

Instalación de una instancia de AWS	Página 2
Instalación de Magento y dependencias	Página 4
Instalación y configuración del tema	Página 12
Importación de productos, categorías y colores	Página 15
Modificaciones de diseño	Página 18
Guía de estilos	Página 26
Hacer la web multilingüe	Página 27
Migración, Restauración y Workflow	Página 28

Instalación de una Instancia AWS

En primer lugar nos debemos crear una cuenta de Amazon Web Service (AWS).

1. Una vez logueado, abra la consola de Amazon EC2 en <https://console.aws.amazon.com/ec2/>.
2. En el panel de la consola, seleccione lanzar instancia (Launch Instance).
3. Elija una imagen de máquina de Amazon (Amazon Machine Image (AMI)) en la página, que muestra una lista de las configuraciones básicas, que sirven como moldes para la instancia. En mi caso he seleccionado ubuntu. Observe que este AMI está marcado como "libre o free".
4. En la elección del tipo de instancia, puede seleccionar la configuración de hardware de la instancia. Seleccione el t2.micro tipo, que es seleccionado por defecto. Tenga en cuenta que este tipo de instancia es elegible gratuitamente.

Nota:

Las Instancias T2, como por ejemplo t2.micro, deben ponerse en marcha en una VPC. Si su cuenta de AWS soporta EC2-Clasic y no tiene una VPC en la región seleccionada, el asistente de puesta en marcha crea una VPC para usted y puede continuar con el siguiente paso. De lo contrario, la revisión y el botón de lanzamiento está desactivado y hay que elegir siguiente: Configurar detalles Instancia y siga las instrucciones para seleccionar una subred.

5. Elija revisión y puesta en marcha para dejar que el asistente complete las otras opciones de configuración para usted.
6. En la revisión y lanzamiento de Instancia, en grupos de seguridad, se verá que el asistente crea y se selecciona un grupo de seguridad para usted. Se puede utilizar este grupo de seguridad (en mi caso tuve que editarlo posteriormente para seleccionar las IPs y puertos permitidos), o, alternativamente, se puede seleccionar el grupo de seguridad que ha creado al conseguir crear usando los siguientes pasos:
7. Elija editar los grupos de seguridad.

- a. Para Configurar un Grupo de Seguridad, asegúrese de que selecciona un grupo de seguridad existente.
- b. Seleccione su grupo de seguridad de la lista de grupos de seguridad existentes y elija Revisar y ejecutar.
- c. En la revisión Instancia Lanzamiento página, elija lanzamiento.

8. Cuando se le pida un par de claves, seleccione Elegir un par de claves existentes, a continuación, seleccione el par de claves que ha creó previamente.

Como alternativa, puede crear un nuevo par de claves. Seleccione Crear un nuevo par de claves, introduzca un nombre para el par de claves, y luego seleccione Descargar par de claves. Esta es la única oportunidad para que guarde el archivo de clave privada, así que asegúrese de descargarlo. Guarde el archivo de clave privada en un lugar seguro. Usted tendrá que proporcionar el nombre de su par de claves cuando se inicia una instancia y la clave privada correspondiente cada vez que se conecte a la instancia.

Precaución:

No seleccione el proceder sin un par de claves. Si inicia la instancia sin un par de claves, entonces no se puede conectar con ella.

Cuando esté listo, seleccione la casilla de verificación, y luego elegir Lanzamiento de Instancia.

9. En la página de confirmación se le permite saber que la instancia está poniendo en marcha. Elija Ver Instancias (View Instances) para cerrar la página de confirmación y volver a la consola.
10. En la pantalla de Instancias, puede ver el estado de la puesta en marcha. Se tarda un poco de tiempo en lanzar una instancia. Cuando se inicia una instancia, su estado inicial pendiente. Cuando se inicie la instancia, su estado cambia a corriendo y recibe un nombre DNS público. (Si el DNS Público (IPv4) está oculto, seleccione el botón Mostrar / Ocultar en la esquina superior derecha de la página y luego seleccione Public DNS (IPv4)).

La preparación de la instancia puede tardar un poco para que pueda conectarse a ella. Compruebe que la instancia ha pasado sus comprobaciones de estado; se puede ver esta información en la columna comprobaciones de estado.

*Para conectarte al servidor, en la página de instancias, seleccionando la instancia a la que nos queremos conectar, solo debemos pulsar el botón Conectar, y se nos proporcionarán los comandos para los distintos métodos de acceso.

En mi caso uso PuTTY para conectarme por ssh:

-Host Name: ubuntu@ec2-52-39-40-120.us-west-2.compute.amazonaws.com

-fichero que contiene la clave: antonio.pem

Teniendo el servicio ssh nos bastaría con usar el siguiente comando:

```
ssh -i "antonio.pem" ubuntu@ec2-52-39-40-120.us-west-2.compute.amazonaws.com
```

Documentación oficial:

http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/EC2_GetStarted.html

Instalación de Magento y tecnologías necesarias para su funcionamiento.

Paso 1 - Instalar Nginx

- Acceder al servidor (por ejemplo, mediante SSH) y actualizar el repositorio.

```
sudo su  
apt-get update
```

- A continuación, instalar Nginx:

```
apt-get install -y nginx
```

- Verificar que nginx se ha instalado correctamente comprobando el puerto:

```
netstat -plntu | grep 80
```

Paso 2 - Instalar y configurar PHP-FPM

En este paso vamos a instalar PHP 7 en el modo PHP-FPM. Además vamos a instalar las siguientes extensiones de PHP que son requeridas por Magento.

```
php-gd
```

```
php-mhash
```

```
php-mcrypt
```

```
php-XSL
```

```
php-pear
```

```
php-jabón
```

- Instalar los paquetes con el siguiente comando:

```
apt-get install php7.0-fpm php7.0-mcrypt php7.0-curl php7.0-cli php7.0-mysql php7.0-gd php7.0-xsl php7.0-json php7.0-intl php-pear php7.0-dev php7.0-common php7.0-mbstring php7.0-zip php-soap libcurl3 curl -y
```

- Ahora editar los archivos php.ini para fpm y cli.

```
nano /etc/php/7.0/fpm/php.ini  
nano /etc/php/7.0/cli/php.ini
```

- y aumentar límite de memoria y el tiempo máximo de ejecución de php y activar la compresión zlib añadiendo las siguientes líneas al final de los archivos:

```
memory_limit = 512M
max_execution_time = 1,800
zlib.output_compression = On
```

- Guardar y salir del editor.
- Reiniciar el servicio de PHP-FPM para aplicar los cambios de configuración:systemctl restart php7.0-fpm

Paso 3 - Instalar y configurar MariaDB

- Voy a utilizar MariaDB en lugar de MySQL:

```
apt-get install mariadb-server mariadb-client -y
```

- Establecer la contraseña de usuario root MariaDB con este comando:

```
mysqladmin -u root password mypassword
```

```
mysql_secure_installation
```

```
Set root password? [Y/n] Y
```

```
New password:
```

```
Re-enter new password: <-- Enter the new password
```

```
Remove anonymous users? [Y/n] Y
```

```
... Success!
```

```
Disallow root login remotely? [Y/n] Y
```

```
... Success!
```

```
Remove test database and access to it? [Y/n] Y
```

```
Reload privilege tables now? [Y/n] Y
```

```
... Success!
```

- A continuación, conectar a la consola de MySQL (la consola de MariaDB se inicia con el comando *mysql*) con la contraseña de root, cree una base de datos con el nombre '*magentodb*' y un usuario '*magentouser*' con la contraseña '*magentouser@*'.

Entrar al shell MySQL:

```
mysql -u root -p
```

- En el shell MySQL, ejecutar estos comandos:

```
create database magentodb;
```

```
create user magentouser@localhost identified by 'magentouser@';
```

```
grant all privileges on magentodb.* to magentouser@localhost identified by
'magentouser@';
flush privileges;
\q
```

```
MariaDB [(none)]> create database magentodb;
Query OK, 1 row affected (0.00 sec)

MariaDB [(none)]> create user magentouser@localhost identified by 'magentouser@';
Query OK, 0 rows affected (0.00 sec)

MariaDB [(none)]> grant all privileges on magentodb.* to magentouser@localhost identified by 'magentouser@';
Query OK, 0 rows affected (0.00 sec)

MariaDB [(none)]> flush privileges;
Query OK, 0 rows affected (0.00 sec)

MariaDB [(none)]> \q
Bye
root@magento:~#
```

- Base de datos creada y configurada.

Paso 4 - Instalar y configurar Magento 2

Vamos a instalar Magento en el directorio '/var/www/magento2'. Para la instalación de Magento, necesitamos el composer de PHP.

- Instalar php composer

- Ir al directorio raíz, descargar el archivo de instalación del composer con curl y ejecutarlo para instalar el composer.

```
cd ~/
curl -sS https://getcomposer.org/installer | php
```

- Mover el archivo 'composer.phar' al directorio bin del servidor y cambiar el nombre para que pueda ser ejecutado fácilmente:

```
mv composer.phar /usr/bin/composer
```

- Ahora verificar que el comando de composer está funcionando:

```
composer -v
```

- Descarga y extracción de Magento 2

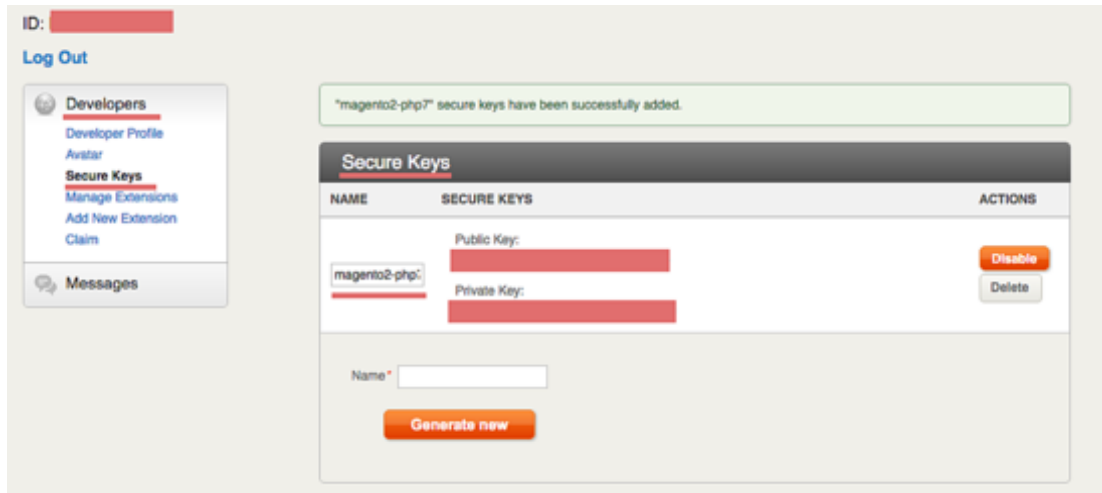
- Una vez que tenemos el fichero comprimido de Magento (<https://magento.com/tech-resources/download>, adjunto en la documentación), lo descomprimos en el directorio '/var/www/magento2'

```
cd /var/www/
mkdir magento2
cd magento2
tar -xzf Magento-CE-2.1.5-2017-02-20-05-36-16.tar.gz
```

- Hecho.

- Configurar la clave de Magento

- Crea una cuenta en el sitio web de Magento repo.magento.com . Esta cuenta es necesaria para utilizar Magento y el composer de Magento. Cuando te registras, hay que ir a la pestaña 'My Account > Developer > Secure Keys' y genera tus claves.



- Instalación de componentes para Magento

- Ir al directorio de instalación de Magento 2 '/ var / www / magento2' y ejecutar el comando:

```
cd /var/www/magento2/
composer install -v
```

- Se pedirá para la autenticación de Magento, utilizar la clave pública como nombre de usuario y la clave privada para la contraseña.



- Configurar el Virtualhost de Nginx

Magento ofrece una configuración de virtualhost de Nginx ya hecha, por lo que sólo hay que incluirla en nuestra configuración.

- Ir al directorio del host virtual de Nginx y crear un nuevo archivo llamado magento:

```
cd /etc/nginx/sites-available/
nano magento
```

- pegar la siguiente configuración:

```
upstream fastcgi_backend {
    server unix:/run/php/php7.0-fpm.sock;
}

server {

    listen 80;

    server_name magento.zedcloud.es;

    set $MAGE_ROOT /var/www/magento2;

    set $MAGE_MODE developer;

    include /var/www/magento2/nginx.conf.sample;
}
```

- Guardar y Salir.
- Añadir el dominio magento.zedcloud.es es en el fichero /etc/hosts
- Ahora activar el host virtual y reiniciar Nginx:

```
ln -s /etc/nginx/sites-available/magento /etc/nginx/sites-enabled/
systemctl restart nginx
```

- Instalar Magento

- Vamos a instalar Magento a través de la línea de comandos. En el directorio de Magento '/ var / www / magento2 /' existe un archivo binario con el nombre 'Magento' que se utiliza para instalar y administrar Magento. Ejecutar el comando:

(En mi caso antes de ejecutar el comando de instalación he dado permisos 777 'sudo chmod 777 -R /var/www/magento2' a todas las carpetas y subcarpetas de magento).

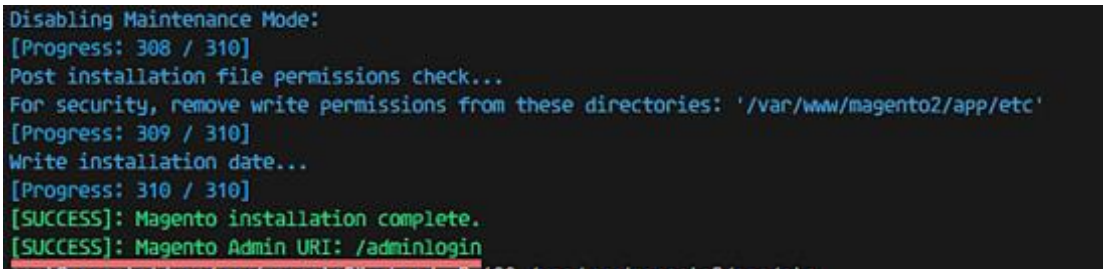
```
bin/magento setup:install --backend-frontname="admin_login" \
--base-url=http://127.0.0.1/ \
--db-host="localhost" \
--db-name="magentodb" \
--db-user="magentouser" \
--db-password="magentouser@" \
--language="es_ES" \
--currency="EUR" \
--timezone="Europe/Madrid" \
--use-rewrites=1 \
```



```
--use-secure=0 \  
--base-url="http://localhost" \  
--base-url-secure="https://localhost" \  
--admin-user=antonioadmin \  
--admin-password=antonio3393 \  
--admin-email=antonio@zoocha.es \  
--admin-firstname=antonio \  
--admin-lastname=admin \  
--cleanup-database \  
--session-save=db \  
--sales-order-increment-prefix="ORD$"
```

backend-frontName = la página de administración de nuestro sitio Magento, utilizamos 'admin_login'.

base-url = asegurarse de que es la misma que la configuración del host virtual.



```
Disabling Maintenance Mode:  
[Progress: 308 / 310]  
Post installation file permissions check...  
For security, remove write permissions from these directories: '/var/www/magento2/app/etc'  
[Progress: 309 / 310]  
Write installation date...  
[Progress: 310 / 310]  
[SUCCESS]: Magento installation complete.  
[SUCCESS]: Magento Admin URI: /adminlogin
```

- Al final del procedimiento de instalación deberían verse estas líneas:

[SUCCESS]: Magento installation complete.

[SUCCESS]: Magento Admin URI: /adminlogin

- Antes de probar la instalación de Magento, asegurarse de que el propietario del directorio web es 'www-data', a continuación, reiniciar nginx.

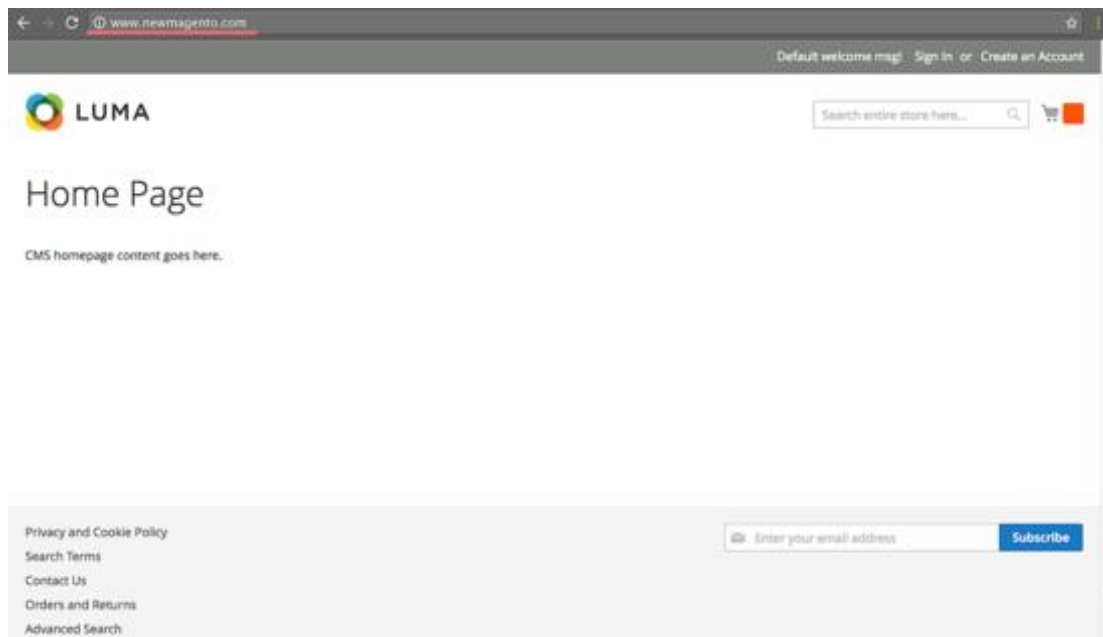
En mi caso he seguido dando permisos 777 a toda la rama de magento para evitar problemas (Más adelante proporcionaremos los permisos necesarios para que la web sea totalmente segura.)

```
cd /var/www/magento2/  
chmod 700 /var/www/magento2/app/etc  
chown -R www-data:www-data .
```

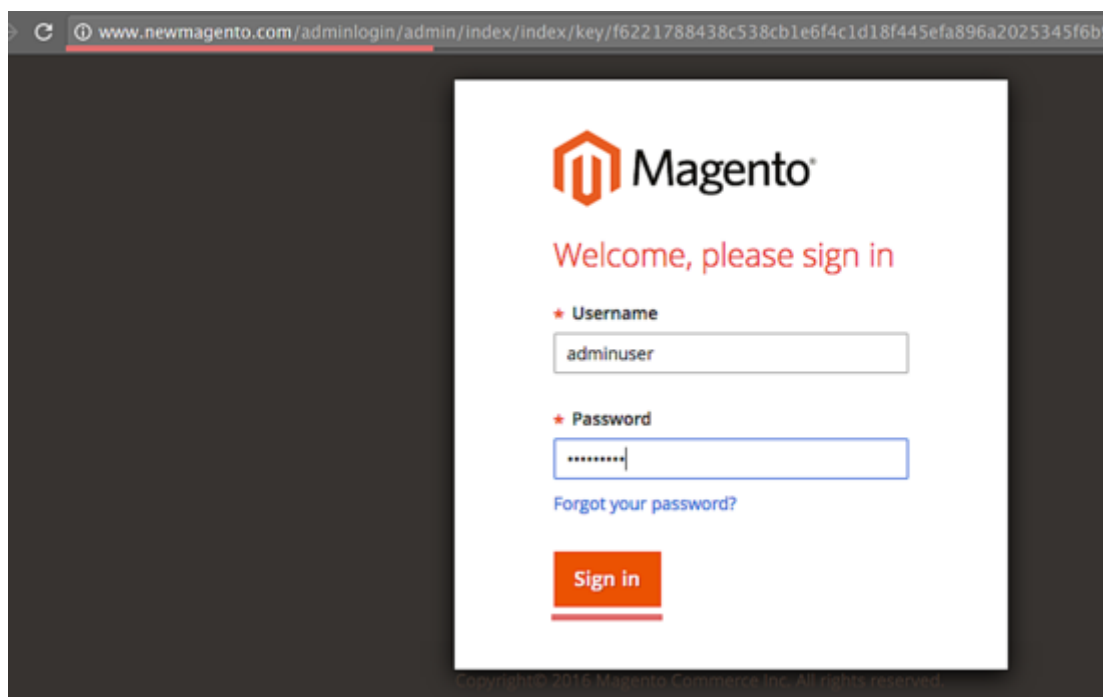
```
systemctl restart nginx
```

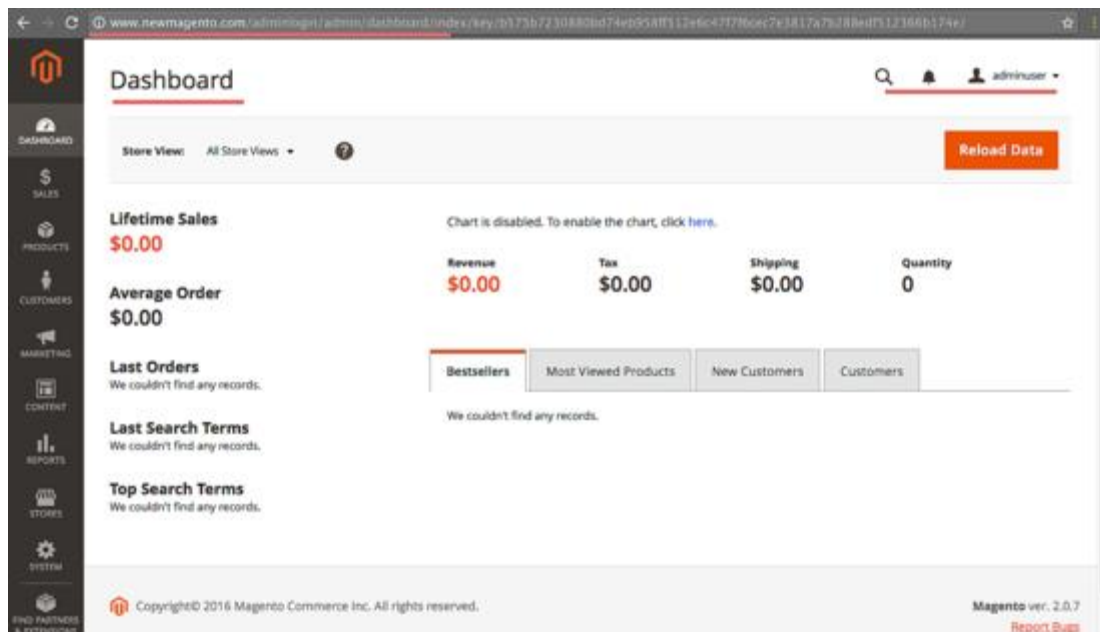
- Ahora escribir el dominio de Magento en el navegador:

En este caso, el nombre de dominio es: magento.zedcloud.es.



Ahora comprueba que puedes acceder y navegar por la parte de administración de magento a través de la url: magento.zedcloud.es/admin_login (en mi caso)





Documentación:

<https://www.howtoforge.com/tutorial/how-to-install-magento-with-nginx-on-ubuntu/>

-Posible error

En una de las instalaciones que he hecho de magento, me surgió un error (no podía navegar por la parte de administración una vez logueaba con la cuenta de admin).

Aporto la url en la que encontré la solución para ese error por si se diese el caso:

<http://gotechnies.com/css-javascript-files-loading-magento-2-installation/>

Instalación del tema

Ahora vamos a instalar el tema (adjunto en la documentación) para lo que hemos seguido la propia documentación que el tema que se encuentra en la carpeta documentation.

- Una vez que tenemos descomprimida la carpeta del tema, debemos copiar el contenido de la carpeta theme-files/cleversoft-ione-v1.0.0 (app y pub) en la carpeta donde tenemos magento(/var/www/magento2)
- Seguidamente ejecutamos el comando de composer mediante línea de comandos:

```
cd /var/www/magento2
```

```
composer update -v
```

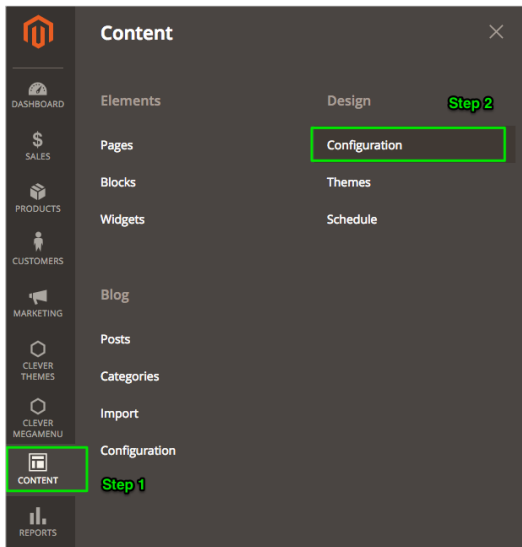
- Ejecutamos el comando de magento para actualizarlo con los ficheros del tema:

```
php bin/magento setup:upgrade
```

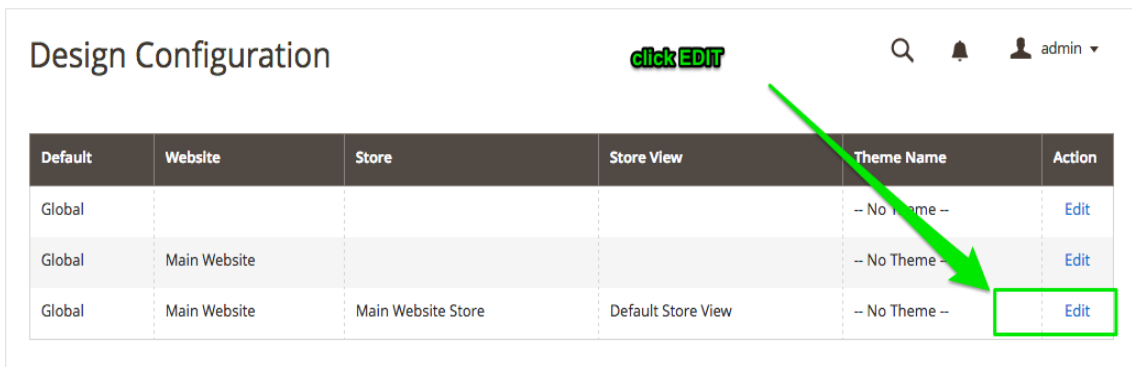
- Una vez finaliza la actualización, se nos pide que recompilemos magento. Para ello usamos el siguiente comando:
- `php bin/magento setup:di:compile`
- Establecemos permisos 777 de nuevo para evitar conflictos.
- Ejecutamos el siguiente comando para importar todo el contenido estático que el tema contiene:

```
php bin/magento setup:static-content:deploy
```

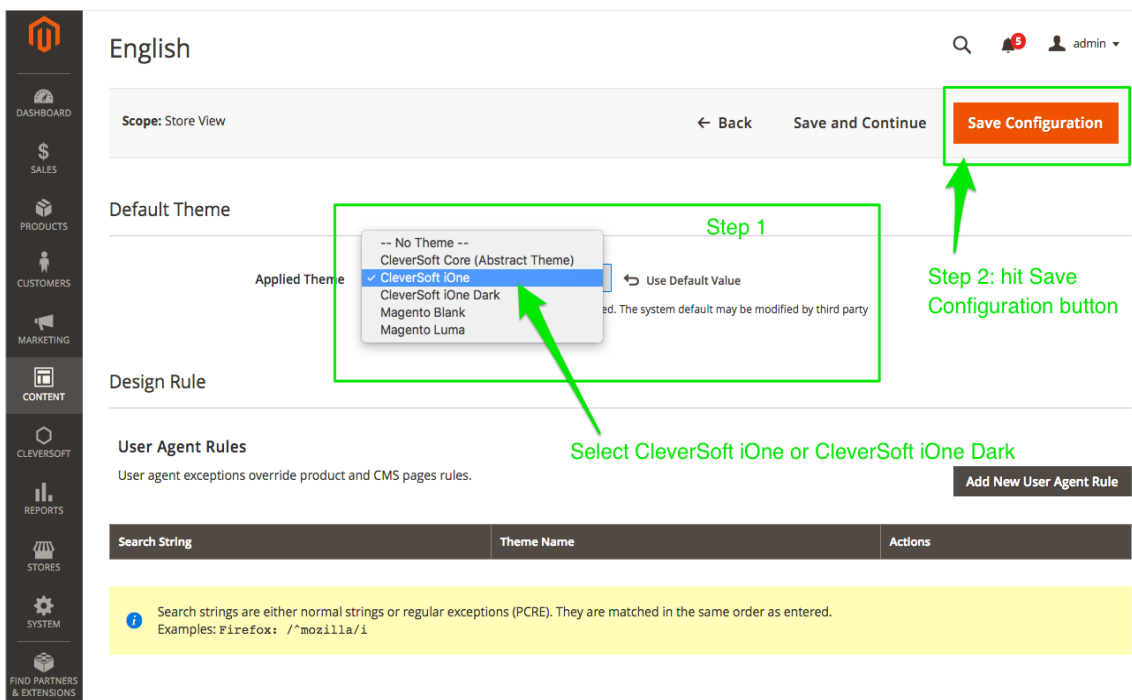
- Volvemos a establecer todos los permisos a la rama de magento
- Accedemos a la parte de administración de nuestra web
`magento.zedcloud.es/admin_login`
- *Nos dirigimos a la configuración de contenido:*



- Editamos nuestro sitio principal:



- y escogemos el tema que queremos aplicar:



- Guardamos la configuración y si se nos muestra un mensaje diciendonos que hay que refrescar la cache nos dirigimos a System/Cache Management y clickeamos sobre Flush Magento cache.
- Una vez hecho esto, debemos deshabilitar la opción Blocks Html output y refrescar toda la cache, en el mismo lugar en el que administramos la cache:

Cache Management

Flush Cache Storage Flush Magento Cache

Step 2: Refresh All Cache

Refresh Submit 13 records found (13 selected)

step 1: disable Blocks HTML output

Cache Type	Description	Tags	Status
<input checked="" type="checkbox"/> Configuration	Various XML configurations that were collected across modules and merged	CONFIG	ENABLED
<input checked="" type="checkbox"/> Layouts	Layout building instructions	LAYOUT_GENERAL_CACHE_TAG	ENABLED
<input checked="" type="checkbox"/> Blocks HTML output	Page blocks HTML	BLOCK_HTML	DISABLED
<input checked="" type="checkbox"/> Collections Data	Collection data files	COLLECTION_DATA	ENABLED
<input checked="" type="checkbox"/> Reflection Data	API interfaces reflection data	REFLECTION	ENABLED
<input checked="" type="checkbox"/> Database DDL operations	Results of DDL queries, such as describing tables or indexes	DB_DDL	ENABLED
<input checked="" type="checkbox"/> EAV types and attributes	Entity types declaration cache	EAV	ENABLED
<input checked="" type="checkbox"/> Customer Notification	Customer Notification	CUSTOMER_NOTIFICATION	ENABLED
<input checked="" type="checkbox"/> Page Cache	Full page caching	FPC	ENABLED
<input checked="" type="checkbox"/> Integrations Configuration	Integration configuration file	INTEGRATION	ENABLED
<input checked="" type="checkbox"/> Integrations API Configuration	Integrations API configuration file	INTEGRATION_API_CONFIG	ENABLED
<input checked="" type="checkbox"/> Translations	Translation files	TRANSLATE	ENABLED
<input checked="" type="checkbox"/> Web Services Configuration	REST and SOAP configurations, generated WSDL file	WEBSERVICE	ENABLED

- Seguidamente, usaremos la opción de importar todo el contenido del tema (Páginas de sistema de gestión de contenidos y bloques estáticos) que se encuentra en Cleversoft Theme Settings en el apartado de Installation. Para hacerlo tenemos que seleccionar 'Import Static Blocks, Import CMS Pages, seleccionar una versión Demo, en mi caso Home 6 Fashion English, e importarla. Yo también he escogido las opciones de Overwrite Existing Blocks and Pages para tener contenidos de ejemplo y poder hacer modificaciones sobre ellos. Terminado todo esto, guardamos toda la configuración y refrescamos cache (opción Flush Magento cache explicada anteriormente)

Importación de productos, categorías y colores

Ahora debemos importar nuestros propios productos y categorías para poder modificar los contenidos que el tema trae por defecto y adaptarlos a nuestros productos:

En este proyecto, estoy usando los productos de la web <https://oceansrattanfurniture.com/>, una web que ha realizado la empresa para un cliente hace varios años con drupal, y que quieren actualizar y modernizar con esta tecnología que estoy usando.

Para obtener los productos de esta web he accedido a la url <https://oceansrattanfurniture.es/google-feed-uk> y he descargado un csv con todos los productos con la opción 'csv' situada en la parte inferior izquierda de la página.

Ahora toca filtrar y formatear todos los productos para que puedan ser importados en magento, para lo que hay que seguir los siguientes pasos:

1. descargar un csv de ejemplo de magento para realizar importaciones de productos y conocer los parámetros que debo incluir en el nuevo csv. (En el CMS de Magento accedemos a System/Import y seleccionamos productos donde nos encontramos con el enlace para descargar un csv de ejemplo de importación de productos.)
2. Estudiar detenidamente el csv de oceans en el que podemos observar que cada imagen de un producto se trata como un producto individual, es decir, si tenemos un producto con 5 imágenes distintas, se crean 5 productos con todos los parámetros idénticos a excepción del identificador (en el que a través de distintos códigos numéricos se hace referencia a un color) y las urls de las imágenes.
3. Investigar la forma de crear 'productos configurables', que son productos genéricos que llevan asociados varios productos simples relacionados por un atributo común (en mi caso el atributo color).

Siguiendo la siguiente documentación me creé un producto configurable de prueba (<https://www.mageplaza.com/kb/how-to-configure-swatches-in-magento-2.html>) que luego me exporté a un csv para ver los atributos requeridos. (System/Export y tras seleccionar productos, en la parte inferior de la página le damos a continuar)

4. Con toda esa información y a base de consultar muchas funciones de php y hacer muchas modificaciones sobre el script en php que he creado(consultando un ejemplo de script para transformar ficheros csv que el jefe me ha proporcionado) para transformar el csv de oceans en un csv apto para la importación en magento y para importar todas las imágenes de los productos de oceans en nuestro servidor, el script ha quedado tal cual podemos ver en el script adjunto a la documentación (product_sync_prueba.php) y el csv que se genera al ejecutarlo también está adjunto (magento_upload.csv)
5. Antes de realizar la importación tenemos que añadir los valores necesarios para el atributo color (los colores de los productos de oceans) en magento:

En el CMS Accedemos a Stores/Attributes/Product.

Seleccionamos el atributo color, establecemos tipo Dropdown y añadimos los siguientes valores:

Manage Options (Values of Your Attribute)

Is Default	Admin*	Default Store View		
<input type="radio"/>	BLACK	Black	Settings	Delete
<input type="radio"/>	COFFEE	Coffee	Settings	Delete
<input type="radio"/>	MOCHA	Mocha	Settings	Delete
<input type="radio"/>	LATTE	Latte	Settings	Delete
<input type="radio"/>	NATURE	Natural	Settings	Delete
<input type="button" value="Add Option"/>				

También podemos añadir valores a un atributo con un script que he creado y adjuntado a la documentación (functions.inc). Cuando creamos webs con más de 20 colores o incluso queremos crear varios atributos y asignarle valores es mucho más útil y rápido realizar esta tarea con un script.

6. Una vez tenemos todo preparado, en el CMS accedemos a System/Import, seleccionamos productos y elegimos tipo Add/Update y seleccionamos nuestro csv magento_upload.csv.
También podemos realizar el import con el siguiente comando en la carpeta de nuestro proyecto:
php bin/magento catalog:import [-i|--images_path=["..."]] csv_file

7. Finalmente, tras hacer la importación nos aparecerá un mensaje diciéndonos que hay que reinicializar los índices y refrescar la cache, para lo que debemos realizar lo siguiente:

En la consola de comandos:

```
cd /var/www/magento2  
php bin/magento indexer:reindex
```

Refrescar la cache a través del CMS como hemos explicado anteriormente o usando el comando:

```
php bin/magento cache:flush
```

-Posible error:

Un posible error que puede surgir tras la importación de productos es que magento por sí solo busque las imágenes en cache y aún no estén cacheadas. Para ello debemos ejecutar el siguiente comando:

```
php bin/magento catalog:images:resize
```

Modificaciones sobre el diseño de la web

- En primer lugar, vamos a añadir el menú de navegación para acceder a las distintas categorías y regresar a la página principal:

Para ello en el CMS accedemos a Cleversoft/Create New Menu

1. Primero asignamos un título y un identificador para el menú.
 2. Seguidamente, en el apartado de Edit Menu creamos los enlaces (Custom Links) que deseemos, rellenando los campos de Title (título de la página) y URL (url de la página que ya está establecida por defecto y podemos verlas en el apartado del CMS Products/Categories), y los vamos añadiendo al menú hasta que deseemos guardar la configuración del menú.
 3. Una vez finalizado el menú debemos acceder a Content/Elements/Blocks y seleccionar Mega Menu, en donde tenemos que seleccionar el identificador del menú que hemos creado y guardar la configuración. Tras esto refrescamos la cache y podremos ver nuestro menú en la página inicial de la aplicación.
- En segundo lugar vamos a modificar los colores de la cabecera usando los colores y el logotipo de la web de oceans. Para ello tenemos que acceder a Stores/Configuration y acceder al apartado de Cleversoft Theme.
 1. En settings, dentro del apartado header en las 3 imagenes del logo asignamos la imagen del logo de oceans que nos hemos descargado.
 2. En design tenemos que modificar el background color del apartado Header, en el cual pondremos el color #000000, y el background color y sticky header del apartado main menu en los cuales estableceremos el color #91804d.
 - Ahora toca modificar los contenidos de la página principal y establecer el diseño de las páginas de las categorías de productos.

Al haber importado y sobrescrito los bloques de contenido del tema que hemos instalado, si accedemos al apartado del CMS Content/Element/Pages y seleccionamos editar la home, dentro del apartado content podremos ver que hay distintos bloques añadidos, en los cuales me he fijado para hacer los míos propios.

Para crear o modificar un slider existente, lo hacemos en el cms desde la opción Cleversoft/Clever Slider/Sliders Manager

Para crear o modificar un bloque lo hacemos desde content/blocks.

Ejemplo de código de mi bloque de ofertas de la home:

```
<div class="container">
  <div class="row">
    <div class="col-xs-12 discount-products-block" style="min-height: 500px;">
      <div class="col-xs-12 col-sm-3 discount-text" style="min-height: 500px; display:
flex; flex-direction: column; justify-content: center; text-align: center;">
        <p style="font-size: 35px;">EXTRA</p>
        <p style="font-size: 45px;">10%</p>
        <p style="font-size: 45px;">OFF</p>
        <p style="font-size: 35px;">SALE PRICES</p>
        <p style="font-size: 25px;">ENDS SOON!</p>
      </div>
    <div class="col-xs-12 col-sm-9" style="padding: 0;">
      <div class="col-xs-12 imgs-1-2" style="min-height: 240px;">
        <div class="col-xs-6 img-1" style="min-height: 240px; background-image:
url('{{media url="wysiwyg/img-1.jpg"}}'); background-position: center center;
background-size: cover; background-repeat: no-repeat;"></div>
        <div class="col-xs-6 img-2" style="min-height: 240px; background-image:
url('{{media url="wysiwyg/img-2.jpg"}}'); background-position: center center;
background-size: cover; background-repeat: no-repeat;"></div>
      </div>
      <div class="col-xs-12 img-3" style="min-height: 250px; background-image:
url('{{media url="wysiwyg/img-3.jpg"}}'); background-position: center center;
background-size: cover; background-repeat: no-repeat;"></div>
    </div>
  </div>
</div>
```

Para añadir nuestro css custom accedemos a Cleversoft/Clever Themes v1.2.3/Custom css.

Mi css custom es el siguiente ("app/design/frontione/web/css/custom.less"):

```
body{
    font-family: 'Lato', sans-serif !important;
}
```

```
/*language switcher*/
```

```
.switcher-language, .switcher-label{
```

```
        color: #ffffff !important;
    }
    .switcher-language{
        float: right !important;
    }
}
```

```
/*promos message*/
```

```
.zoo-promo-msg , .zoo-promo-msg:hover{
    background-color: #b40303 !important;
    color: white !important;
}
```

```
/*Header*/
```

```
body .page-wrapper #zoo-main-content .container .main_content_area .new-arrivals
.zoo-product-collection01 .block .block-content .products-grid .product-items .owl-
stage-outer .owl-stage .owl-item .product_hover .product-item-info .zoo-inner-
product-item-info .product-item-details .product-item-name .product-item-link {
    color: #999999 !important;
}
@media (max-width: 767px){
    .page-header .header-content {
        background: #91804d !important;
    }
}
@media only screen and (max-width: 767px){
    .page-header {
        background-color: transparent !important;
    }
}
@media (min-width: 768px){
    .clever-horizontal-menu .clever-mega-menu li.level0>a.menu-link {
        margin: 0 !important;
    }
}
```

```
/*main slider*/
```

```
.my-slider{
    margin-top: 4rem;
}
.flex-active-slide .zoo-slideshow-text-content h1.slideshow__title.align-center {
    text-shadow: 1px 1px 3px #000000;
    width: 60%;
    margin-left: 20%;
}
.zoo-slideshow-text-content .zoo-button-slide a {
    box-shadow: 1px 1px 3px #000000;
}
```

/*categories carousel*/

```
#cat-carousel {
    margin: 50px 0 50px 0 !important;
}
#cat-carousel .description {
    background: #c7b299;
    padding: 1rem;
}
#cat-carousel .description .name-cat {
    color: white;
}
img.img-cat {
    height: auto !important;
}
```

/*discount products block*/

```
.discount-products-block{
    margin-top: 6rem;
}
.discount-text{
    background-color: #b40303 !important;
}
.discount-text p{
    color: #ffffff !important;
    margin: 0px;
}
.img-1{
```

```

        border-right: 5px solid white;
    }
    .img-2{
        border-left: 5px solid white;
    }
    .imgs-1-2{
        padding: 0;
        border-bottom: 10px solid white;
    }

```

```

/*new arrivals carousel*/

```

```

body .page-wrapper #zoo-main-content .container .main_content_area .new-arrivals
.zoo-product-collection01 .zoo-main-heading .zoo-heading-wrapper-title h3 span{
    font-size: 30px;
}
body .page-wrapper #zoo-main-content .container .main_content_area .new-arrivals
.zoo-product-collection01 .block .block-content .products-grid .product-items .owl-
dots {
    bottom: 45px !important;
}
body .page-wrapper #zoo-main-content .container .main_content_area .new-arrivals
.zoo-product-collection01 .block .block-content .products-grid .product-items .owl-
stage-outer .owl-stage .owl-item .product_hover .product-item-info .zoo-inner-
product-item-info .zoo-product-image a img {
    width: 100%;
    text-align: center;
    height: auto;
    max-height: 140px;
}

```

```

/*Category page Main image*/

```

```

.page-wrapper .category-view .category-image img {
    max-height: 400px;
}
.catalog-category-view .page-title-wrapper, .zoo-product-collection01 .page-title-
wrapper{
    display: initial !important;
}

```

```
/*Quick view product*/
```

```
.product-items .product-item-info .zoo-product-image .options-slideup {  
    background: #c7b299 !important;  
}  
.product-items .product-item-info .zoo-product-image .options-slideup:hover {  
    background: black !important;  
}  
.options-slideup .zoo-quickview {  
    color: white !important;  
}  
.products-grid .product-item-info {  
    height: 300px !important;  
}  
.product-items .product-item-info .zoo-product-image a img.product-image-photo {  
    max-height: 150px;  
}
```

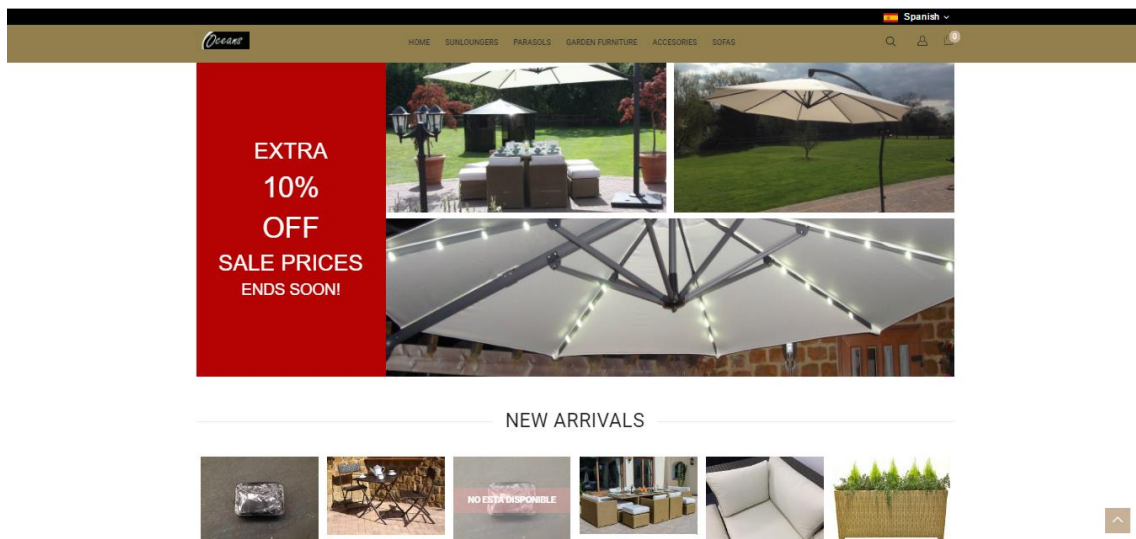
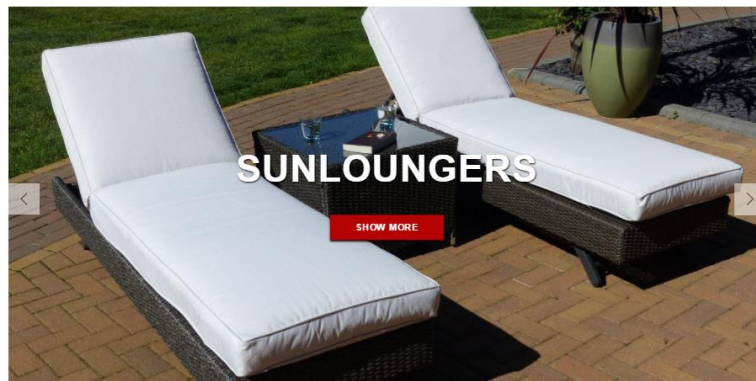
```
/*html product*/
```

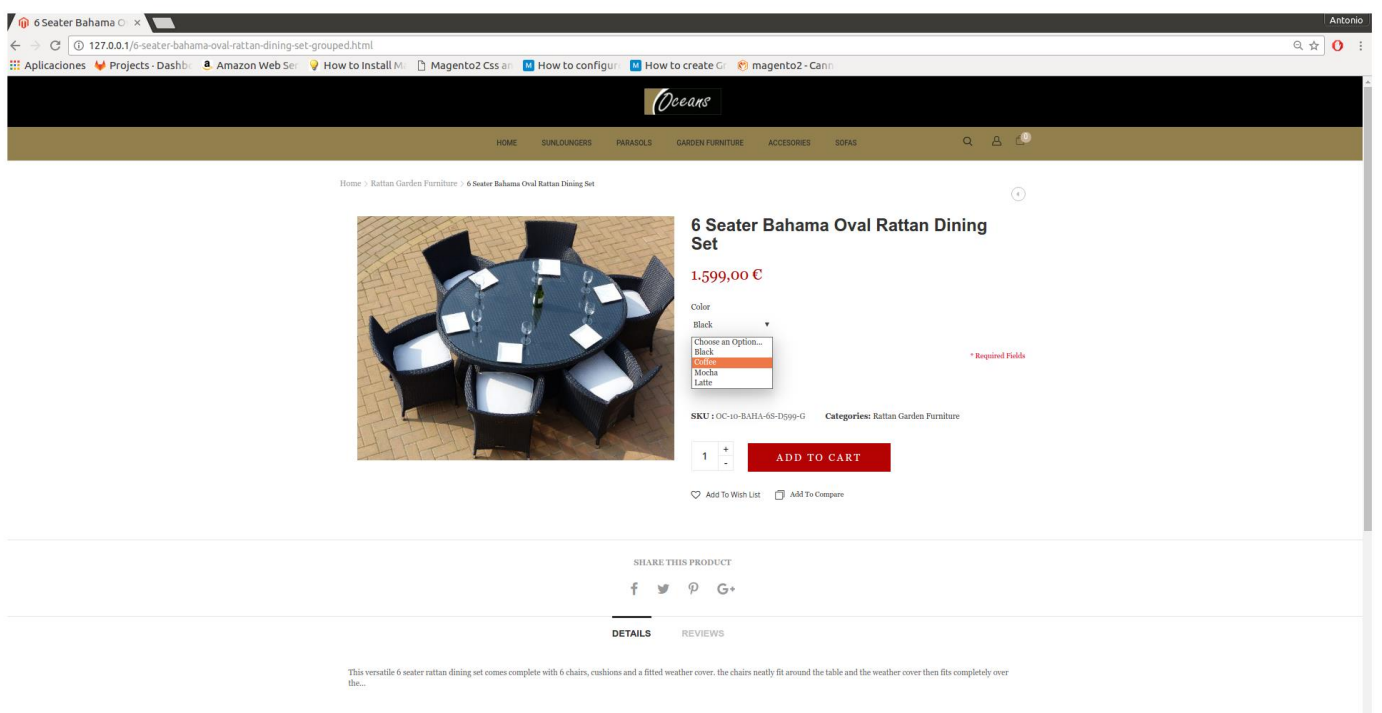
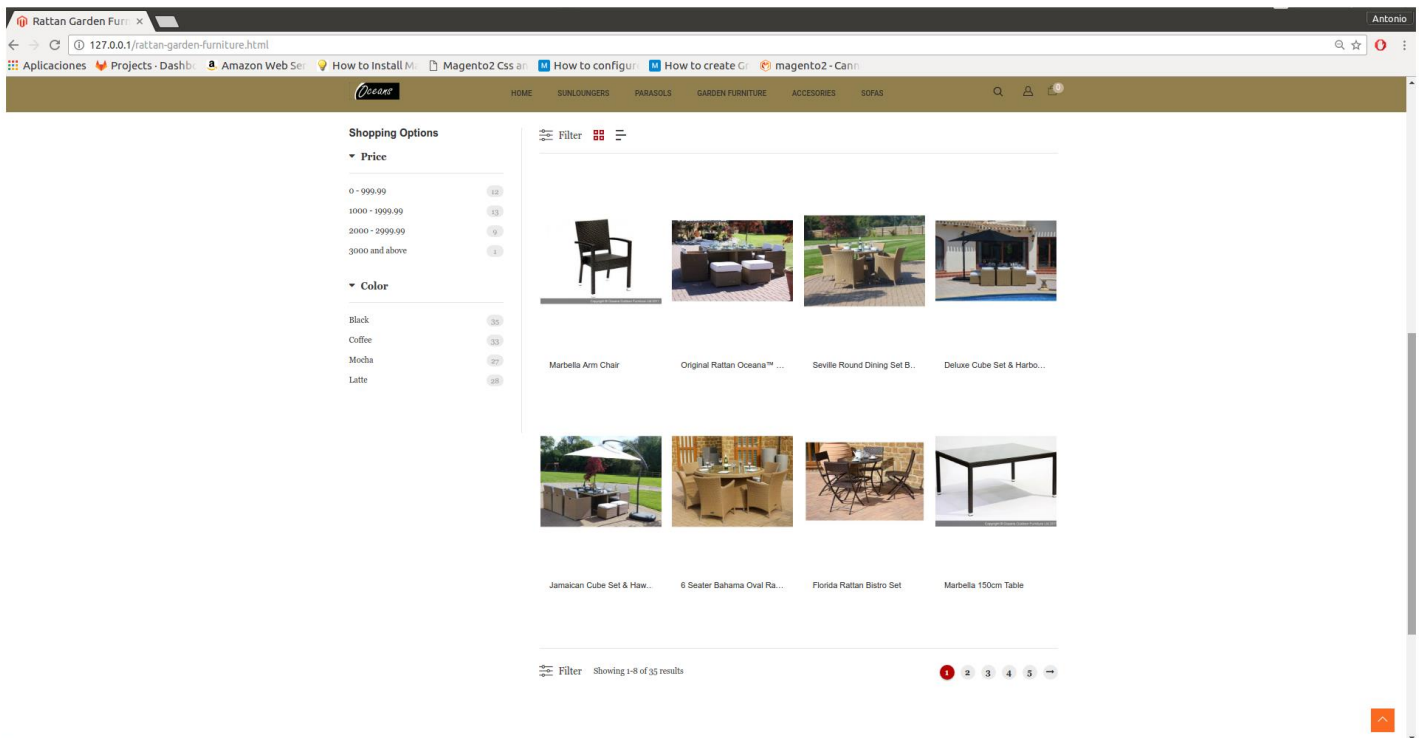
```
@media (min-width: 992px){  
    .product-add-form p.required {  
        margin-top: -20px;  
        right: 100px;  
    }  
}  
.product-info-main .product-info-price .price-box .price-final_price>span.price-  
wrapper, .product-infomain .product-info-price .price-box .price-  
final_price>span.price-wrapper {  
    margin-top: 2rem;  
}  
.catalog-product-view #product_addtocart_form .box-tocart {  
    margin: 4rem 0 7rem;  
}  
.product-info-main .stock.available {  
    margin: 10px;  
}  
.zoo-product-meta {  
    padding: 10px 0;  
}
```

```
/*footer*/
```

```
.page-wrapper .copyright {  
  text-align: center;  
}
```

Haciendo modificaciones de las categorías y diseño de los bloques por defecto y añadiendo mis propios bloques custom he conseguido este diseño:





-Posible error

En mi caso, trabajando con magento en local, en vez de en el servidor, me di cuenta de que no podía añadir productos al carrito desde todas las páginas de la aplicación. Tras mucho buscar

encontré un post en el que se hablaba de un error al trabajar en local y establecer como url base localhost en vez de 127.0.0.1, por lo que tuve que realizar las siguientes modificaciones sobre la base de datos del proyecto.

```
UPDATE core_config_data  
SET value = 'http://127.0.0.1/'  
WHERE path IN ('web/secure/base_url', 'web/unsecure/base_url');
```

```
UPDATE core_config_data  
SET value = NULL  
WHERE path = 'design/head/includes';
```

Guía de estilos

¿Qué estándares se han seguido para establecer el diseño de nuestra web? Esta web se dedica a la venta de muebles de rattan entre los que predominan los colores marrones y negros, por ello estos son los colores que más predominan en el diseño junto con el color rojo que se usa para resaltar información:

La fuente que se ha usado en toda la web es la Lato, una fuente bastante común y estándar para que el usuario pueda leer todo con claridad

La cabecera y el pie de página tendrán un color de fondo negro, que hará resaltar las letras de color blanco como las del logo de la web. También serán los únicos elementos que ocuparán todo el ancho de la pantalla.

El menú de navegación tendrá un fondo marrón claro con los enlaces en color negro.

Seguidamente pasamos al contenido en sí de la web, en el cual todo el fondo será de color blanco y las letras de colores negros o grises.

Los botones normalmente tendrán un fondo marrón con letras en negro que al hacer hover pasarán a tener un fondo negro con las letras en blanco.

¿Cuándo usaremos el color rojo? Para las cosas que queremos resaltar. Como nuestros anuncios de descuentos o los botones de añadir productos a nuestro carrito.

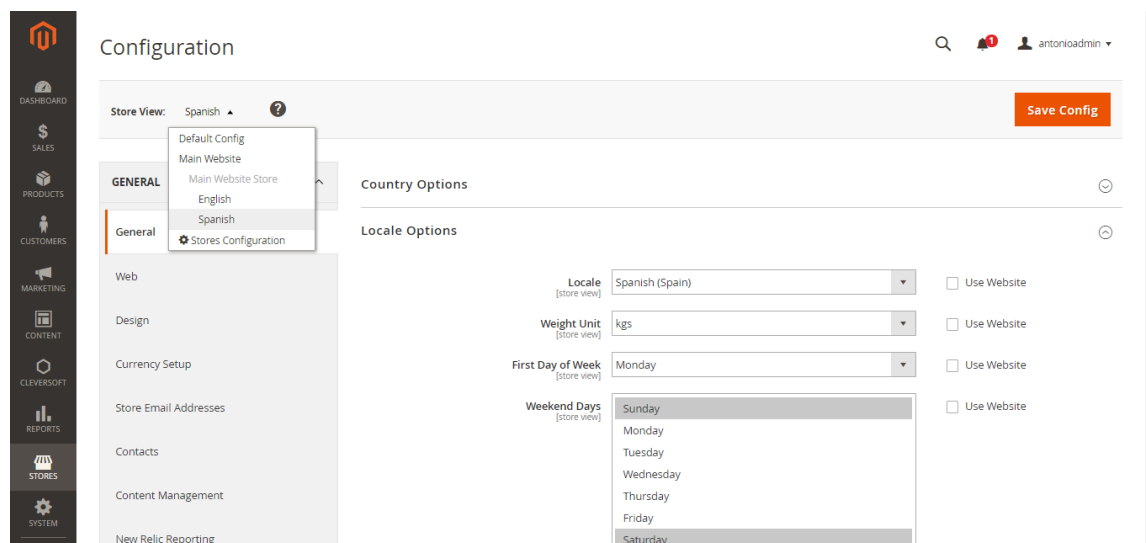
Hacer nuestra web multiligüe

1. En primer lugar tenemos que descargarnos o instalar el paquete de la traducción española de magento que lo hacemos a través del composer:

```
composer require magento2translations/language_es_es:dev-master
php bin/magento cache:clean
```

Una vez instalado el paquete deberíamos poder ver el csv de traducción en `/app/design/frontend/CleverSoft/ione/i18n/es_ES.csv`, sobre el cual podemos acceder modificaciones e insertar nuevas traducciones.

2. En segundo lugar tenemos que crearnos un nuevo store “almacen o sitio” al cual le aplicaremos el mismo tema que al sitio por defecto.
Al sitio por defecto le tenemos que asignar en el store view el nombre del idioma por defecto “inglés” y al nuevo sitio tenemos que asignarle “español”
3. Seguidamente modificaremos las opciones de idioma del nuevo sitio. Para ello accedemos a Stores/Settings/Configuration/General/General/Locale Options. En la parte superior izquierda seleccionamos el store view sobre el que queremos realizar cambios, en nuestro caso el español, y en locale seleccionamos Spanish.



4. Una vez realizado esto y guardados los cambios solo nos queda incluir (si no está incluido) un selector de idioma en la web. En mi caso, accediendo a `app/design/frontend/CleverSoft/ione/CleverSoft_Base/templates/html/cases` y modificando el layout que nosotros tengamos establecido en el diseño del header “en mi caso es el 1), dentro de header-content he añadido la siguiente línea:

```
<?php echo $this->getChildHtml("store_language"); ?>
```

5. Hecho esto solo tenemos que limpiar caché y podremos ver nuestro selector en la home. Dicho todo esto, hay que añadir que de esta forma no se puede traducir todo el contenido completo de la web, ya que existen bloques estáticos como el menú o mi bloque custom de productos en oferta que están escritos directamente en html. La solución para ello es duplicar páginas o bloques y asignarlas a un store view concreto.

Migración, Restauración y Workflow

Se supone que cualquier persona que realiza una web, desea que esta pueda ser migrada a otro servidor en cualquier momento, o incluso tenerla en cualquier local para realizar sus propias pruebas.

- Una forma sencilla de migrar una web en magento es usando el siguiente comando:
php bin/magento setup:backup -v --code --media --db
El cual nos genera una copia de seguridad del código, una de todas las imágenes o videos y otra de la base de datos dentro de la carpeta:
carpeta del proyecto/var/backups
En el pc que queramos tener este proyecto debemos instalar nginx y php y configurarlos de la forma que se ha explicado anteriormente. Seguidamente creamos la base de datos e importamos la copia de seguridad a través del comando:
mysql -u usuario -ppassword nombre_bd<backup_bd
Tras hacer esto debemos acceder a la base de datos y cambiar la url:

```
UPDATE core_config_data  
SET value = 'http://127.0.0.1/'  
WHERE path IN ('web/secure/base_url', 'web/unsecure/base_url');
```

La url debe coincidir con la que hayamos configurado en nginx. Hecho esto, solo queda tener una instalación limpia de magento con nuestro servidor web apuntando a dicha ruta y descomprimir dentro nuestras backups.

También hay que tener en cuenta que dentro de nuestra carpeta del proyecto, en el fichero /app/etc/env.php los parámetros de conexión con nuestra base de datos son correctos.

- Teniendo nuestras copias de seguridad, una forma de restaurar nuestra web es a través del siguiente comando en nuestra carpeta raíz del proyecto:
`php bin/magento setup:rollback -c code_backup -m media_backup -d database_backup`
- Por último, cuando estamos trabajando con webs que se encuentran publicadas, la mejor forma de trabajar y modificar cosas sobre ella es haciendo pruebas primero en un servidor de prueba “stager” desde el cual se hará push a un repositorio una vez que las pruebas sean satisfactorias.
 Si por cualquier motivo las pruebas salen mal, siempre podemos hacer un pull de ese repositorio, ya que contendrá siempre el código del servidor real “live”.
 En cambio si las pruebas son satisfactorias, se hará un push al repositorio y seguidamente desde el servidor real se hará un pull, con lo cual el código quedará actualizado.
 Siempre que se modifica código en el servidor real, hay que ejecutar el siguiente comando, que por desgracia necesita poner la web en modo mantenimiento durante un par de minutos:

```

sudo bin/magento maintenance:enable
sudo rm -rf var/cache var/di var/generation var/view_preprocessed pub/static
sudo php -f bin/magento cache:clean
sudo php -f bin/magento cache:flush
sudo php -f bin/magento cron:run
sudo php -f bin/magento indexer:reindex
sudo php -f bin/magento setup:upgrade
sudo php -f bin/magento setup:di:compile
sudo php -f bin/magento setup:db-data:upgrade
sudo php -f bin/magento setup:db-schema:upgrade
sudo php -f bin/magento setup:db:status
sudo chmod -R 777 .
sudo php -f bin/magento indexer:reindex
sudo php -f bin/magento deploy:mode:set developer -s
sudo php -f bin/magento setup:di:compile
sudo php -f bin/magento setup:static-content:deploy es_ES en_US
sudo chmod -R 777 .
sudo service nginx restart
sudo bin/magento maintenance:disable

```

Los comandos necesarios para convertir nuestra carpeta del proyecto en un repositorio local son los siguientes:

```
git init
```

```
git remote add origin remote repository URL  
# Sets the new remote  
git remote -v  
# Verifies the new remote URL
```

Una vez hecho esto, modificamos el fichero .gitignore a nuestro gusto.
En nuestro caso, solo sería necesario subir al repositorio la carpeta app.
Ya solo queda decir que los comandos para guardar cambios en el repositorio son:

```
git add .  
git commit -m "First commit"  
git push origin master
```

Y para descargar cambios del repositorio:

```
git pull origin master
```